Evaluating an Evolutionary Approach for Reconstructing Gene Regulatory Networks

Dion J. Whitehead[1†], Andre Skusa[1†] and Paul J. Kennedy[2]

[1] NRW Graduate School in Bioinformatics and Genome Research, Center of Biotechnology (CeBiTec), University of Bielefeld, Postfach 10 01 31, D-33501 Bielefeld, GERMANY

[2] Faculty of Information Technology, University of Technology, Sydney, PO Box 123, Broadway, NSW 2007, AUSTRALIA

[†] Both authors contributed equally

andre.skusa@cebitec.uni-bielefeld.de

Abstract

Reconstructing networks from (partial) incomplete data is a general problem in biology. We use an evolutionary approach in an artificial network creation and reconstruction framework to investigate limitations of gene expression network inference from simulated microarray data. For this, the simulated dynamics of the evolved networks are optimized to fit the target dynamics. Evolving networks with similar dynamics is not as difficult as comparing the resulting network topologies to the original network to be reconstructed.

Introduction

A common problem in biology is the reconstruction of networks from incomplete data, for example biochemical networks (Arkin et al., 1997). An important example of this problem is the inference of gene regulatory networks (GRNs) from time series of gene expression data. GRNs may be modelled using a variety of formalisms of differing complexities ranging from directed graphs, boolean networks, differential equations and Bayesian networks (De Jong, 2002).

Time series gene expression data from high throughput assays (eg. cDNA microarray experiments (Baldi and Hatfield, 2002)) may be used to recover the original GRNs that generated the data. This is called a *reverse problem*. The reverse problem has been approached in many different ways, depending on the formalism used to model the GRN. For example, (Liang et al., 1998) infer GRNs modelled using random boolean networks (RBN); (Bower and Bolouri, 2000) used hierarchical clustering on a boolean model; (Kikuchi et al., 2003) find weights for systems of differential equations using genetic algorithms (GA) and (Friedman et al., 2000) infer Bayesian network representations of a GRN.

Microarray experiments (Parmigiani et al., 2003) are a recent cellular biology technology able to measure the level of expression of thousands of genes in cells at one instant. There are two kinds of experiments: cDNA microarray experiments that compare the relative gene expression levels of two samples on the same chip, and oligonucleotide microarrays (Affymetrix) that use one chip per sample.

Microarray data is difficult to use for modelling networks of gene expression. There are challenges processing the data itself, partly due to the difficulty of comparing data from different samples and partly because of the large number of genes (tens of thousands) compared to the number of repetitions of experiments (tens), which is often referred to as the high dimensionality of microarray data. See (Huber et al., 2003) for a recent model of the error associated with microarray experiments and (Yang et al., 2000) for details on normalising data within and between experiments. However, the biggest problem using microarray data to build models of gene expression networks is not the noise of the sample, nor the intrinsic noise of the biological phenomena. It is the fact that microarrays only measure one part of the "real" network, that is, the transcription stage of gene regulation, ignoring RNA regulation, post–translational modification, localization signals, phosphorylation, etc. Noise starts to look unimportant against all the missing data.

A challenge when inferring GRNs is that it is difficult to assess the effectiveness of algorithms because the actual GRNs are mostly unknown. Some researchers approach this challenge by comparing the induced GRNs with known GRNs for the same data. See, for example, (Wahde and Hertz, 2000).

This difficulty has led some researchers, eg. (Repsilber and Kim, 2003), (P. Mendes and Ye, 2003), to test proposed methods with artificially generated, but plausible GRNs. Microarray data is simulated for the artificial GRN and the proposed solution to the reverse problem is compared with the known artificial GRN.

This contribution follows a similar approach but looks at a slightly different question: what are the similarities and differences between artificial GRNs that generate the same gene expression data time series.

Our methodology consists of four stages (illustrated in Fig. 1): (i) create an artificial GRN (called the *target GRN*) at random including marked mRNA and gene nodes, (ii) simulate microarray data for the target GRN, (iii) evolve artificial GRNs which produce simulated microarray data matching the simulated microarray data for the target GRN, and (iv) compare the evolved GRNs with the target GRN.

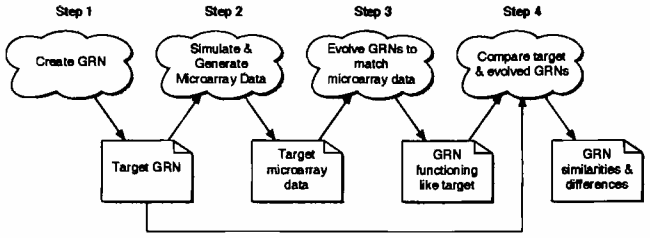

Figure 1: Methodology used in this paper

Network Model

GRNs consist of many different types of reactions, but with current technologies only a subset of these reactions can be directly measured. In the case of microarray experiments, only mRNA levels can be directly measured. Crucial measurements that are not made are the actual concentration of the protein for which the mRNA is translated, the resulting variant if different exon combinations are possible, post-translational modification, protein localization, its current state (for example bound by an inhibitor protein) and many others, as eg. the recent discovery of the impact of small RNA strands (*microRNA*) on all levels of genomic regulation (Ambros, 2001). The result is a network with many unknown nodes and edges (Fig. 2). This is the basis for our inquiry: assuming it is *possible* for a network to be reconstructed from experimental data, what affect do invisible nodes have on the reconstructed networks?

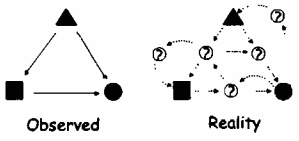

Figure 2: Example of incomplete measurement. Dark nodes are mRNA levels measured by microarray technologies. Unknown nodes are the unmeasured entities.

In most models of GRNs the nodes represent equivalent entities ie. gene products. The GRNs we use are different in that nodes represent all of the entities that regulate protein production (eg. genes, mRNA, modified protein states etc.) rather than simply gene products. This generalisation is motivated by the fact that gene regulation also occurs at places other than just promoter sites.

In our model, based on microarray experiments, nodes are assigned to one of two categories: *visible* ie. experimentally measured or *invisible* ie. not measured. Visible nodes represent entities that may be detected in the simulated microarray experiment (ie. mRNA molecules). Conversely, invisible nodes represent molecules undetectable by the simulated microarray. Both kinds of nodes, however, contribute towards the network dynamics.

Method Overview

Step 1: Create Random Gene Regulatory Networks

One possibility of randomly creating artificial regulatory networks (ARNs) is to generate a bit string and mimic the process of gene regulation occurring from this genome (Banzhaf, 2003). The network then emerges from the interaction of the components. In contrast to this we generate the network topology directly. Though it is trivial to create networks at random it is not as straightforward to create networks at random which should "make sense", ie. they should exhibit dynamic behavior that is arbitrary, but reasonable in the domain they are created for. In our case, reasonable GRNs are those which do not stop expression at an early time step or where activity is confined to only a small subset of the whole network. Therefore, we applied a method to prune and modify random networks to produce interesting dynamics:

First, a directed random network is created by applying the Erdös-Renyi approach (Erdös and Renyi, 1960) to directed graphs: for a number of nodes each possible edge is tested and drawn if a random number is lower than a given probability p. Each edge is assigned a uniform random weight in the range of $[-1, 1]$. The node which reaches the most other nodes is determined and set to the *starting node* for the simulation (for the concept of a starting node see also *Step 2* in this section). All nodes unreachable by the first node are connected at random to the reachable nodes (also chosen at random).

All nodes in the network are now reachable from the first node in respect to the edge directions, but not necessarily in respect to the reaction rates, because negative reaction rates might inhibit the expression of complete subnetworks "behind" a node. Thus, a second pruning is performed considering the assigned edge weights (which serve later as reaction rates) and all nodes which are not reachable in respect to negative weights are simply deleted from the network. At the end, the nodes are randomly assigned as visible or invisible.

Step 2: Simulate Microarray Data

In this step we simulate microarray data for the target GRN. Simulation of microarray data has been accomplished by other researchers in a variety of ways and with different motivations. Repsilber and Kim (Repsilber and Kim, 2003) start with an artificial GRN and simulate the network dynamics for a number of time steps to build a time series. They then change the initial conditions of the GRN and simulate another time series. A simple model combines the two sets of expression values to give a time series of log ratios. Cui et al (Cui et al., 2003) take a different approach. They use a more complicated model of the error implicit in microarray experiments but generate the raw expression values randomly from statistically plausible probability density functions.

Our approach is more similar to that of Kim and Repsilber. We chose to implement a relatively simple and widespread formalism: ordinary differential equations (De Jong, 2002), based on the so-called Hill function, a sigmoidal function that agrees with experimental data. That the function is non-linear, is important. Even though linear functions are simpler, most biochemical interactions are non-linear, which has important consequences in terms of network dynamics. A degradation term is added to model gradual decay.

$$\frac{dg_i}{dt} = h^+(g_i) - \gamma_i g_i \tag{1}$$

where g_i is the expression level of the ith gene and i ranges from 1 to N for N total genes. $\gamma_i = 0.3$ is the degradation term and $h^+(g_i)$ is the Hill function given by

$$h(g_i) = \frac{x(g_i)^3}{x(g_i)^3 + K} \tag{2}$$

with

$$x(g_i) = \sum_{j=1}^{N} \mathrm{W}_{ji} g_j \tag{3}$$

where W is the matrix of interactions and $K = 1$ the threshold for regulatory influence on g_i. One can imagine the rows representing the nodes, and the columns represented the interaction, such that w_{ij} represents the interaction from node i to node j. To allow for repression of genes, then if $x(g_i)$ is negative then $h^+(g_i)$ is replaced by $h^-(g_i) = 1 - h^+(g_i)$ and K changes sign.

Using Euler's method and a fixed step size, the artificial GRN is simulated using numerical integration of the differential equations (ie. equation 1) for a small number of time steps. At the initial time step, the first node (*starting node*) is given a positive value (2 in our simulations) and the value of all other nodes are set to 0. The network is then simulated for s time steps with node values recorded for each time step to form a time series of network activity (s is chosen such that the steady state starts right after that). In an approach inspired by oligonucleotide microarray experiments (eg. Affymetrix experiments), a single gene expression value is generated for each measured node. At this stage, sampling error and other forms of noise are not added to the simulated data. The result of this step is a matrix of values with a row for each measured node and a column for each time step.

Step 3: Evolve Gene Regulatory Networks

Next, GRNs are evolved using an evolutionary computation algorithm to discover networks that produce similar output to the target GRN. To use an evolutionary approach to discover networks, it must be possible to encode each candidate GRN on an artificial genome. In *Step 2* above, we describe how GRNs are represented as a square matrix W of interaction weights. This matrix completely defines the behaviour of a particular GRN from given initial conditions. For this reason, we simply encode each GRN as its matrix W. The top leftmost $n \times n$ sub–matrix (where n is the size of the target matrix) is constrained such that these nodes may not be lost by evolution. These are the visible nodes. Wiring between these nodes, however, is adaptable by evolution. Also, the first node (ie. at W_{11}) has a hard–wired feedback to itself to simplify the evolution task. All other weights, however, are set by evolution.

The artificial evolution begins with a population of randomly created networks 10% larger than the target matrix with weights drawn uniformly randomly from the range [-1, 1]. However, evolution may increase or decrease the number of genes (rows/columns) in the matrix. The evolution protocol consists of a set of network mutations with an associated probability of occurrence at each time step, for example edges may switch nodes or be deleted, weights may be changed by a small amount, nodes can be duplicated. For the experiments in this contribution, we typically used population sizes of 1000 individuals and run lengths up to 6000 generations.

Each member of the population is scored by a fitness function that quantifies how close the output from the GRN corresponding to the genome is to the output of the target GRN. So, for each genome, the GRN is simulated using equations (1), (2) and (3) with the same initial conditions and constant parameters as for the target GRN. Only expression values for the first n genes (ie. the upper leftmost $n \times n$ submatrix matching the target W matrix) are recorded. The distance of the resulting time series of expression values for each of the n genes to the target values is computed using a simple squared error approach:

$$f = \sum_{i=1}^{n} \sum_{t=1}^{T} (g_{it} - y_{it})^2 \tag{4}$$

where f is the fitness of the genome. Evolution seeks to minimise the value f. f sums over each of the n visible genes y_{it} indexed by gene with i and by time with t for each of the T time steps. The value g_{it} is the target gene expression value of gene i at time t.

An elitist strategy is used to manage the population. After calculating the fitness of the population, the weakest 40% of the population is deleted and replaced with the fittest 40%. Genetic operators are then applied to the population with (small) fixed probability. Genetic operators include: node duplication, node deletion, edge creation, edge deletion, edge mutation (ie. changing the weight value) and edge duplication. Genetic operators do not modify the feedback edge to the "initial" node ie. W_{11} nor delete any of the hard–coded visible nodes in the top leftmost $n \times n$ submatrix. Fitness is computed for members of the new population and evolution continues until networks with suitably small fitness are designed or the maximum number of generations is reached.

Step 4: Compare Gene Regulatory Networks

No sophisticated comparison methods were applied in this paper because the evolved networks were much larger and densely connected than expected. Applying and developing meaningful comparison methods is the most important next step. Here we "manually" compared networks only in parts.

Experiments and Results

The goal for the experiments was to evolve networks similar to a given target network regarding the behavior, ie. the expression time series. We chose two target networks for evolution from the randomly created networks, each with 20 nodes. One had a simple sequential connection topology with resulting simple dynamics. The other had a more complicated topology including negative feedback. Both networks contain only visible nodes (see section *Network Model*). However, the evolution protocol allowed the addition and deletion of invisible nodes. By manually comparing the time series of networks of size 20 we concluded that a distance smaller than or equal to approx. 1.0 indicated very similar network dynamics. The same mutation probabilities were used in both experiments. Several parameter settings have been tested. Different simulation lengths (40 and 20 time steps respectively) were used for the two networks to capture the significant dynamics before the steady state is reached. To simplify the plots some significant nodes were selected for drawing the expression level time series.[1]

Simple Sequential Network

The first target we chose can be seen in Fig. 3. It contains mostly sequentially connected genes and therefore the resulting expression time series consists mainly of sequentially increasing expression levels for each gene reaching a final steady state (Fig. 4, top). The main characteristics of this network is the division of the nodes into two groups: one in which the nodes are immediately expressed, and the another where the nodes are expressed around 20 time steps later at much lower expression levels. This reflects the structure of the network, which can be divided into mainly two groups of genes activated separately by the starting gene but then do not subsequently interact. A resulting time series for one of the evolved networks can be found in the bottom of figure 4. The distance to the target network for this was approx. 0.74, and it contains 64 nodes and 826 edges. Comparing the two time series shows that the main characteristics described previously are in the evolved network. Due to the much higher number of nodes and edges in the resulting network, a comparison of the network topology is not trivial and is deferred to future work. The simplicity of the target network topology suggests that a large number of possible topologies could exhibit the target dynamics.

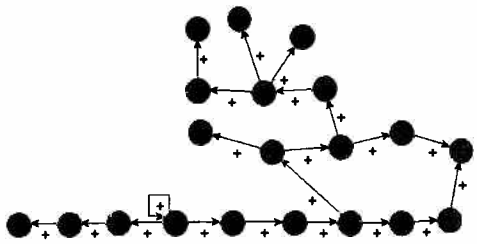

Figure 3: Topology of the simple sequential network. Starting node marked black. The two parts of the network (left and right sides) are arranged according to the two main activity patterns in resulting dynamics (Fig. 4).

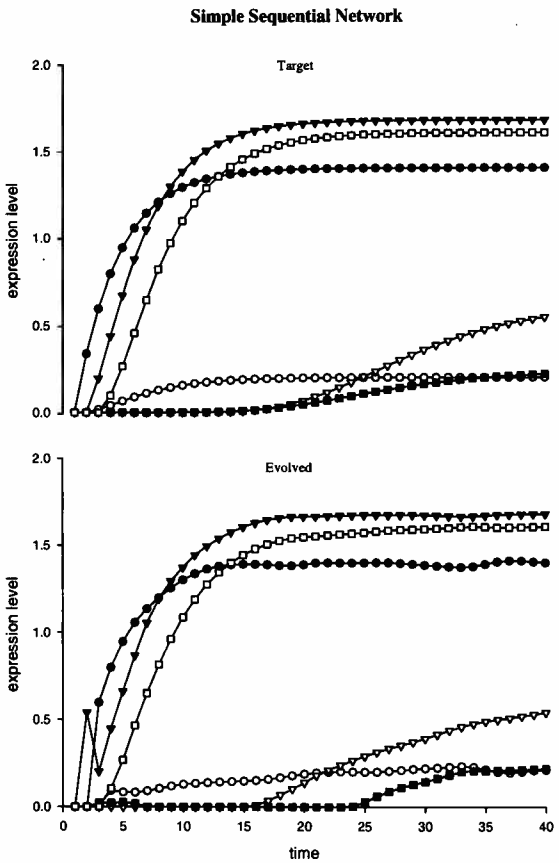

Figure 4: Dynamics of the simple sequential network. Top: Target network behavior. Bottom: Evolved network behavior. Upper and lower expression levels refer to right and left sides in network topology in Fig. 3. Absolute differences between time series points have mean 0.013 and standard deviation 0.0272.

[1] Parameter descriptions and data available on request.

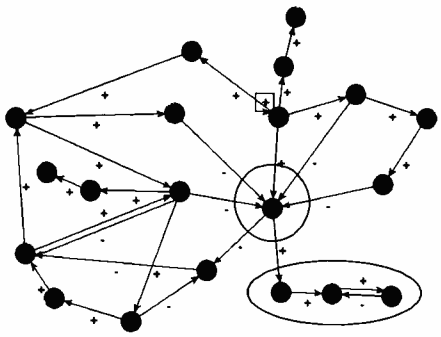

Figure 5: Topology of the network with negative feedback. Starting node is black. Single circled node is crucial to the dynamics and refers to ○ curves in Fig. 7. Circled lower nodes are a motif preserved in evolved network (Fig. 8).

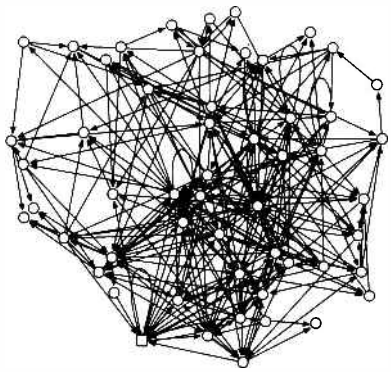

Figure 6: Example of evolved network from the negative feedback network. Network dynamics shown in Fig. 7.

Network with Negative Feedback

The other network chosen for analysis is shown in Fig. 5 and the matching time series at the top of Fig. 7. This network is more complex than the previous due to negative feedback structures leading to delayed inhibition. Dynamics are driven by a node (see Fig. 5) activated by the starting node and inhibited later by all other connecting nodes. The time series shows this node (○ in Fig. 7) initially increasing (by the starting node) then being inhibited to zero, with subsequent nodes also decaying to zero. The time series from one of the evolved networks (Fig. 6) demonstrates that more complicated dynamics can also be approximated (Fig. 7, bottom). The distance between networks is approx. 1.0 and the evolved network consists of 55 nodes and 298 edges. The tendency of networks to grow under evolution is still present, although growth is less than the previous network. This may have been caused by the more restricted space for possible solutions. Comparison of network topologies is non–trivial. Manual examination shows that one substructure in the target network is preserved in the evolved network. The three preserved nodes in the target motif are circled in Fig. 5 and the matching subgraph in the evolved network is shown in Fig. 8. In both cases expression levels of the middle node □ increase after the central node ○ decreases from other influences. Edges in the evolved network motif (Fig. 8) not contributing to these dynamics are not shown. The remaining structure differs only in the position of negative feedback (moved into a self-loop of node □). So, the evolved network preserves not only the overall dynamics, but also some parts of the original topology. Further detailed analysis and development of similarity detection tools is deferred to the future.

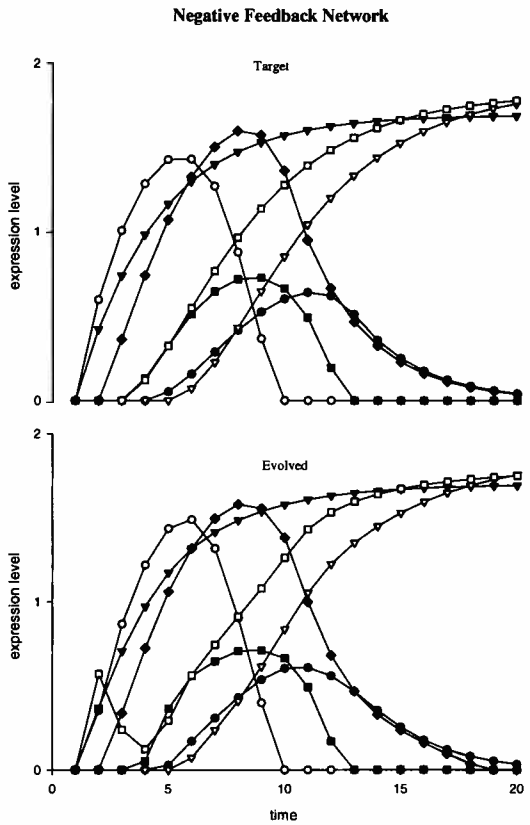

Figure 7: Negative feedback network dynamics. Top: Target. Bottom: Evolved. Delayed inhibition (○ line) belonging to single marked node in Fig. 8 is crucial for network dynamics. Other levels change once the ○ expression decreases (eg. □ or △). Absolute differences between time series points have mean 0.024 and standard deviation 0.0555.

Conclusions and Future Work

Although target and evolved networks have similar dynamics, comparison of topologies is nontrivial. Until structural

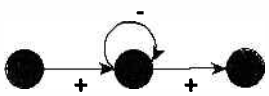

Figure 8: Preserved motif. Edges not contributing to node dynamics removed. Differs by inhibiting edge (self–loop).

(in addition to daynamic) similarity can be measured, results of evolutionary approaches to GRN reconstruction appear unreliable. However, the preserved substructure in the negative feedback network shows future directions. The increase of nodes and edges compared to target networks is a challenge. This may be ameliorated by an evolutionary parsimony constraint, but initial tests showed that this complicated the objective function. Producing a desired behavior by adding nodes is probably easier than initially connecting nodes with optimal reaction rates: nature explores parameter space by gene duplication then modification. Future steps will also complete the stages of our methodology (eg. adjusting the ratio of visible to invisible nodes (ie. the extent of data incompleteness) allows us to tune detection of the artificial GRN by the microarray and adding noise to expression values creates more "realistic" artificial data). Also, other established GRN reconstruction methods will be used.

Acknowledgements

We would like to thank Jacob Köhler for helpful comments. PJK would like to thank Klaus Prank and the NRW Graduate School in Bioinformatics and Genome Research in Bielefeld for hosting his visit and AS and DJW would like to thank the same institution for financial support.

References

Ambros, V. (2001). microRNAs: tiny regulators with great potential. *Cell*, 107:823–826.

Arkin, A., Shen, P. D., and Ross, J. (1997). A test case of correlation metric construction of a reaction pathway from measurements. *Science*, 277(5330):1275–1279.

Baldi, P. and Hatfield, G. W. (2002). *DNA microarrays and gene expression*. Cambridge University Press, Cambridge, UK.

Banzhaf, W. (2003). On the dynamics of an artificial regulatory network. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J. T., and Ziegler, J., editors, *Proc. ECAL 2003, LNAI 2801*, pages 217–227. Springer–Verlag.

Bower, J. M. and Bolouri, H. (2000). *Computational Modeling of Genetic and Biochemical Networks*. MIT Press, Cambridge, MA.

Cui, X., Kerr, M. K., and Churchill, G. A. (2003). Data transformations for cDNA microarray data. *Statistical Applications in Genetics and Molecular Biology*, 2(1).

De Jong, H. (2002). Modeling and simulation of genetic regulatory systems: A literature review. *Journal of Computational Biology*, 9(1):67–103.

Erdös, P. and Renyi, A. (1960). On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci.*, 5:17–61.

Friedman, N., Linial, M., Nachman, I., and Pe'er, D. (2000). Using bayesian networks to analyze expression data. In *Proc. Conf. on Research in Computational Molec. Biology*, pages 127–135, New York. ACM Press.

Huber, W., von Heydebreck, A., Sueltmann, H., Poustka, A., and Vingron, M. (2003). Parameter estimation for the calibration and variance stabilization of microarray data. *Statistical Applications in Genetics and Molecular Biology*, 2(1).

Kikuchi, S., Tominaga, D., Arita, M., Takahashi, K., and Tomita, M. (2003). Dynamic modeling of genetic networks using genetic algorithm and S–system. *Bioinformatics*, 19(5):643–650.

Liang, S., Fuhrman, S., and Somogyi, R. (1998). REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. In *Proc. Pacific Symp. on Biocomputing 3 1998*, pages 18–29.

P. Mendes, W. S. and Ye, K. (2003). Artificial gene networks for objective comparison of analysis algorithms. *Bioinformatics*, 19 Suppl. 2:ii122–ii129.

Parmigiani, G., Garrett, E. S., et al. (2003). The analysis of gene expression data: An overview of methods and software. In Parmigiani, G., Garrett, E. S., Irizarry, R. A., and Zeger, S. L., editors, *The analysis of gene expression data*, pages 1–45, Heidelberg. Springer–Verlag.

Repsilber, D. and Kim, J. T. (2003). Developing and testing methods for microarray data analysis using an artificial life framework. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J. T., and Ziegler, J., editors, *Proc. ECAL 2003*, pages 686–695. Springer–Verlag.

Wahde, M. and Hertz, J. (2000). Coarse–grained reverse engineering of genetic regulatory networks. *Biosystems*, 55:129–136.

Yang, Y., Buckley, M. J., Dudoit, S., and Speed, T. P. (2000). Comparison of methods for image analysis on cDNA microarray data. Technical report 584, Department of Statistics, University of California, Berkeley.